

CONTRACT NUMBER 508830

DEISA
**DISTRIBUTED EUROPEAN INFRASTRUCTURE FOR
 SUPERCOMPUTING APPLICATIONS**

European Community Sixth Framework Programme
RESEARCH INFRASTRUCTURES
 Integrated Infrastructure Initiative

JRA4 – Annual report and Update

Deliverable ID: D-JRA4-7

Due date: April 31, 2007

Actual delivery date: May 28, 2007

Lead contractor for this deliverable: IDRIS – CNRS, France

Project start date : May 1st, 2004

Duration: 4 years

| | | |
|--|--|----------|
| Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006) | | |
| Dissemination Level | | |
| PU | Public | |
| PP | Restricted to other programme participants (including the Commission Services) | X |
| RE | Restricted to a group specified by the consortium (including the Commission | |
| CO | Confidential, only for members of the consortium (including the Commission Services) | |

Table of Content

| | |
|---|----|
| Table of Content..... | 2 |
| 1. Introduction..... | 3 |
| 2. Analysis and adaptation of Molecular Dynamics codes..... | 4 |
| 3. Life Sciences portal applications..... | 10 |
| 4. Perspectives | 18 |

1 - Introduction

The current JRA4 work program involves two different lines of activity:

- Support to research activities in the protein dynamics domain
- Deployment and operation of the initial set of three leading applications that will be operated by the forthcoming Life Sciences portal.

One of the priorities of the DEISA Computational Biology program has been the development of computational tools helping users to access in a more efficient way to DEISA computational facilities, like the MareNostrum system in BSC or the IBM and SGI systems in several other DEISA sites.

Analysis of the needs of scientists working in the area of life sciences has shown that sequence and structure related problems can benefit from supercomputing support. This implies

- The optimization of executable codes to improve their performance on a given platform
- The benchmarking of available programs to determine the best suited code for a given biological problem
- To help the user in the generation of input files in the management of output files as well as in the flow of data between programs.

The protein dynamics work has been guided by the analysis above. For the rest, the JRA4 activity concentrated on the deployment and fine tuning of the three initial basic applications that will be “published” in the Life Science Portal being developed in collaboration with the NICE company.

2. Analysis and Adaptation of Molecular Dynamics Codes protein-protein docking and scoring for the Life Science Portal

Molecular dynamics is one of the most often used techniques in computational biology in supercomputing environments. Molecular dynamics requires a huge amount of computer power as current simulations are still far from the realistic time scales of biological processes. The availability of supercomputer facilities are the only way that this limitation can be overcome. However, molecular dynamics, as most structure based techniques, is normally restricted to experienced users and its current implementations are far from being suitable to be deployed as part of an automated portal. This makes difficult to popularize the technique among general bioinformatics groups, that without the need of performing state-of-the-art simulations, still have structural questions where molecular dynamics would be of great benefit. A possible strategy is to deploy a limited set of standard simulation types with well known parameters. This would not allow fully featured simulations but would cover most of routine operations for those molecular dynamics can be used.

Implementation of the above indicated strategy would require the following steps:

- Choice of a molecular dynamics code with the appropriate characteristics. Previous studies within DEISA project have proposed NAMD from the Univ. of Illinois as the preferred code for a number of reasons. Here we have analyzed NAMD together with two other codes (AMBER and GROMACS) also implemented in MareNostrum in a normal simulation protocols, looking for efficiency and scalability.
- Design of a limited set of simple, standard operations that can be easily run through an automated portal.
- Building of the necessary tools to allow non-experienced users to prepare systems to run NAMD simulations. In a previous report, within this project we have presented the MolDIG suite of scripts allowing fully automated protein preparations. Although MolDIG is suitable for the purposes of the present project, is far too complex for the kind of simple preparations needed, and is based on AMBER formats and parameterization, so modifications are needed to adapt to NAMD, this will focus the forthcoming activities at BSC within JRA4.

2.1 - Benchmark of Molecular Dynamics codes

We have investigated the serial and parallel performance of three second-generation molecular dynamics programs (AMBER v8, NAMD v2.6 and GROMACS v3.2.1) as currently implemented in the MareNostrum supercomputer at the Barcelona Supercomputing Centre. Amber and Gromacs were originally serial programs, subsequently ported to parallel environment in the latest versions, whereas NAMD was originally created with parallel architecture in mind.

Codes have been compiled according to authors recommendations with the necessary adaptations to the architecture, made by Operations Department of BSC. Although no specific improvements of the selected codes have been performed at this stage, some preliminary results suggest that some increase of the performance, specially with NAMD

parallelism, can be achieved with a more aggressive adaptation to specific features of the MareNostrum supercomputer.

To perform the benchmark we selected three proteins of different size with PDB codes 1CQY (starch binding domain of *Bacillus cereus* β amylase), 2HVM (Hevamine A) and 1GND (guanine nucleotide dissociation inhibitor, α isoform).

The sizes of the systems (see Table 1) cover the range of small to large proteins:

| Protein | Protein residues | Protein atoms | Total atoms |
|---------|------------------|---------------|-------------|
| 1CQY | 99 | 1038 | 28310 |
| 2HVM | 273 | 2701 | 31550 |
| 1GND | 430 | 4342 | 66213 |

Table 1: Sizes of the systems used as input to the simulators that we are benchmarking.

All trajectories were collected using “state of the art” simulation conditions using periodic boundary conditions with Particle Mesh Ewald technique to reproduce long range electrostatic effects and keeping constant pressure and temperature.

Serial performance

Benchmarks for the serial version are shown in Figure 1. Amber programs are the slowest in serial execution, with small gains with pmemd over sander, specially over bigger systems. NAMD has similar timings to the Amber ones. Is also faster than pmemd in all cases but its performance is not extremely good. Gromacs is the fastest code, probably due to the special optimization performed at the assembly level in this program.

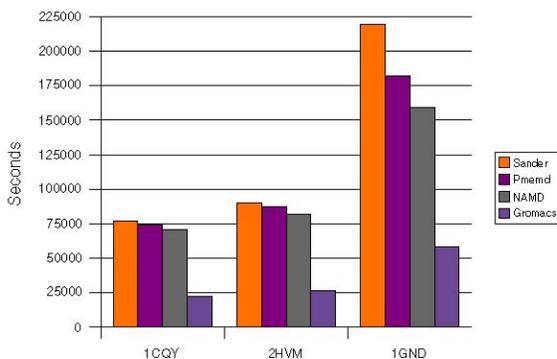


Figure 1: Timings of the different codes running on one processor (wall-clock times)

Parallel performance

Gromacs.

We ran some tests with this simulator, identifying some incoherent results compared with other simulations. This can be seen in Figure 2 and 3, which show artifactual trajectories in GROMACS due to a sharp transition to unfolded structures of proteins

after 1 ns (note the large jump in root mean square deviation in Fig 1), when the protein is found stable with other programs (see AMBER profile in Fig. 2). Analysis of the trajectories discarded the existence of forcefield artefacts, since while OPLS simulations performed in NAMD behave well, those performed in GROMACS unfold the structure. The kind of error reported here was found also in other systems (see MoDEL database at <http://www.bsc.es/>). The time of the trajectory where corruption appears is stochastic and depends on the number of processors: larger the number sooner the corruption. These results lead us towards an issue with the parallelization code, but at this point the reasons for the stochastic behaviour of parallel versions of GROMACS in MareNostrum are unclear, and are being analyzed by GROMACS developers.

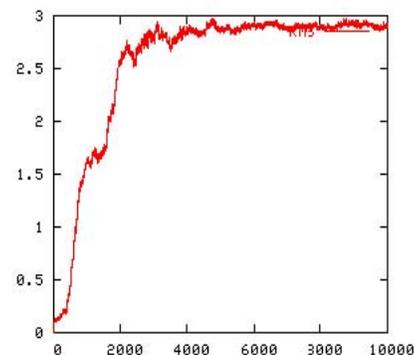


Fig 2. RMSd profile with Gromacs with G43a1 forcefield

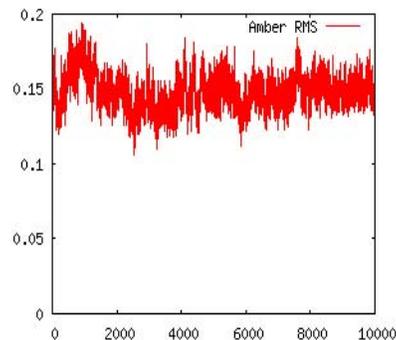


Fig 3. RMSd profile with AMBER and Amber forcefields

No special issues were found in running parallel simulations with AMBER or NAMD. Figures 4 and 5 show Amber parallel performance (fig. 4) and speedup (fig. 5), and so Figures 6 and 7 for NAMD.

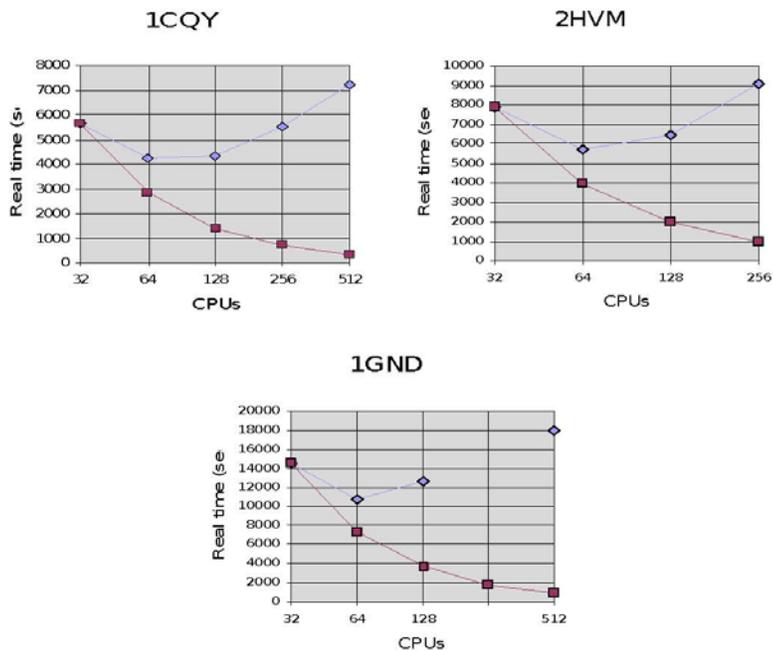


Figure 4. Timings for the simulation of the different solvated proteins with Amber.

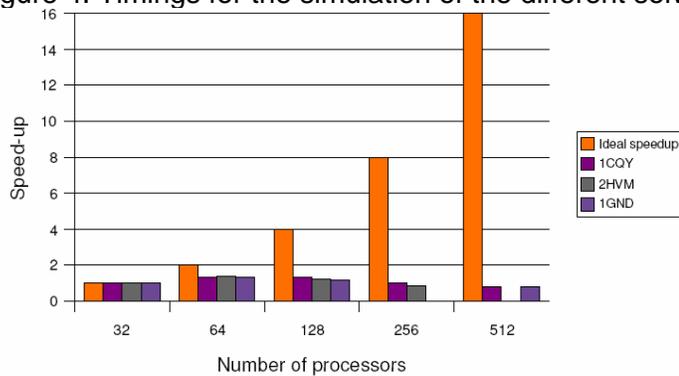


Figure 5. Amber speed-ups.

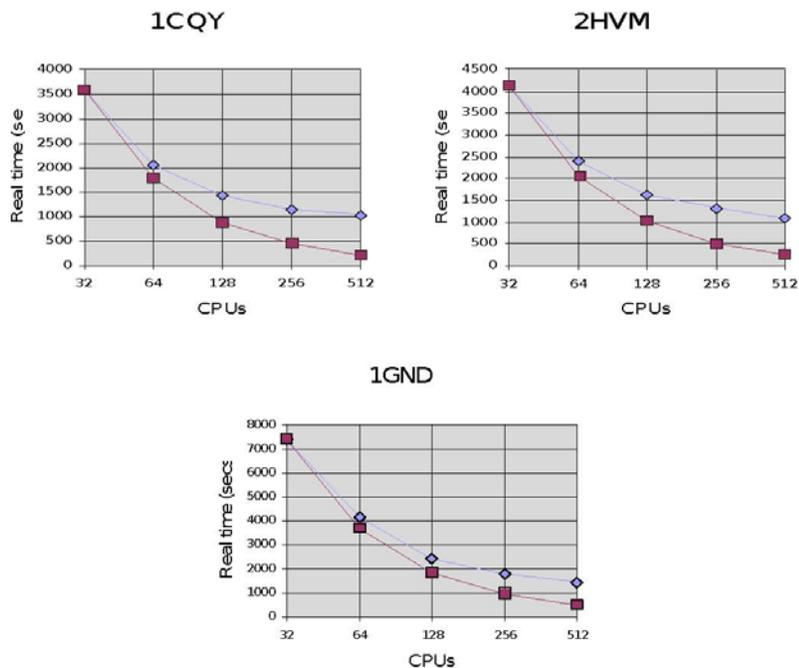


Figure 6. Timings for the simulations of the different solvated proteins with NAMD

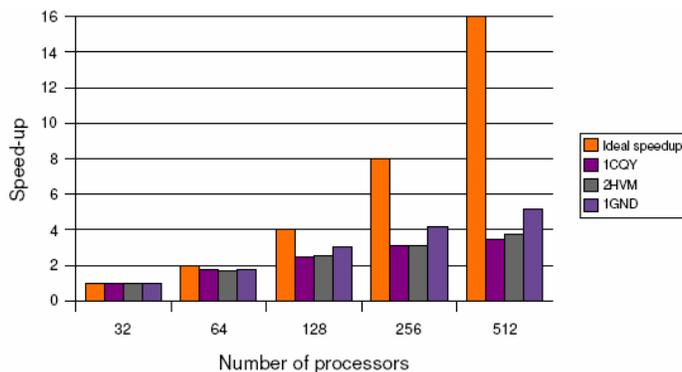


Figure 5. NAMD speed-ups

Given the benchmarks results, it seems quite clear that NAMD is the code with better scalability in MareNostrum architecture and confirms the previous choice of this program to implement within DEISA project. Amber shows an important performance loss when scaling to more than 64 processors. NAMD can keep up its performance because the high number of work units that are generated for each simulation cell, so a better load balancing can be performed, alleviating the problem of having not-working processors waiting for the others to finish. Finally, Gromacs has not been tested, due to the numerical instability detected, but taking into account the method used to distribute data across the processors, which is quite similar to the method used by Amber, an Amber-like behavior can be expected.

2.2 - Standard routine simulations to implement as initial portal services

Two types of simulation cover most needs for non-experienced users: structure optimization and molecular dynamics with explicit solvent and standard conditions. These types of calculation will be offered as follows:

- **Setup step (common).** User will provide a system structure in PDB format. Preparation of the system for simulation with NAMD will be performed using mainly NAMD provided utilities (PSFGEN) and some scripts from the MolDig suite. Setup can be fully automatic for standard DNA or protein structures. However if some special residues (ligands) are involved, they must be prepared manually. User will be offered the possibility of adding special parameters for ligand or modified residues.
- **Calculation type 1.** Structure optimization using two standard minimizers, steepest descents and conjugate gradients. The only parameters needed are the total number of minimization steps and the step to switch optimizers.
- **Calculation type 2.** Molecular dynamics in explicit solvent. Parameters to define (default values in parentheses, suitable for most simulations):

| | |
|--------------------------------|--|
| Type of ensemble: | NVT (constant volume and temperature), NPT (constant pressure and temperature) |
| Ensemble parameters: | Temperature (300K), Pressure (1 Atm) |
| Energy calculation parameters: | Non-Bonded cutoff (8 Å) 1-4 scaling (1.0) Particle Mesh Ewald (PME) for electrostatic calculations (on) Shake (All) |
| Solvent parameters: | Box type (Cube, Octahedron) (Octahedron) Dimensions (X,Y,Z, automatic) (Automatic) Distance (for automatic box) (10 Å) |
| Length of simulation | Timestep (1 fs) Number of steps |
| Output | Format (ASCII) Frequency (1 ps) |

All remaining parameters will have their default values

The proposed form will have the following fields and data formats:

- Input structure (Mandatory). (file, PDB format)
- Added parameters (Optional). (file, NAMD specific format)
- Type of calculation (Mandatory). (Energy minimization | Molecular Dynamics)

All remaining parameters are mandatory with default values.

- Max number of steps. Integer [600]
- Step for Optimizer switch [200] (Minimization only)
- Temperature (K, float) [300.0] (MD only)
- Ensemble. Values NPT, NVT (MD only)
- Pressure (Atm, float) [1.0] (MD only)
- Timestep (fs, float) [1.0] (MD only)
- NB cutoff (Angstroms, float) [8.0]
- 1-4 scaling (float) [1.0]
- PME (on | off) [on]
- Shake (0, H, all) [all] (MD only)

- Box Type (octahedron | cube) [octahedron]
- Dimensions (x,y,z, float | automatic) [automatic]
- Output frequency (ps, float) [1.0] (MD only)
- Output format [ASCII, NMR-PDB, BINPOS] [ASCII]

3 – Life Sciences Portal applications (IDRIS)

One of the activities of JRA 4 is to collaborate with the ESA 3 middleware team to the conception and realization of a Life Science web Portal for DEISA. ESA 3 is dealing with all the system issues of the project and is working with the company NICE owner of a powerful core engine for web portals. Essentially ESA 3 with the help of JRA 4 has to produce very detailed requirements for the company.

3.1 - Conferences with ESA 3

JRA 4, working in close collaboration with ESA 3 has been attending all the video conferences held by the middleware team, trying to answer all the questions in its field and to participate to the discussions keeping the users needs in mind.

3.2 - Choice of applications

For the first version of the portal JRA 4 had decided to start with a limited set of applications, three applications have been chosen after several internal discussions, representing three main fields : genomics, phylogeny and molecular simulations. The choice of portal applications was reported in deliverable D-JRA4-6 (PM30).

3.3 - Installation of the applications

The first three applications have been installed and tested on three sites whose architectures are different from one another as shown on the table below :

| Applications | IDRIS (IBM SP4) | LRZ (Linux SGI) | BSC (PPC) |
|-----------------|-----------------|-----------------|-----------|
| Blast (threads) | X | X | |
| RAxML (MPI) | X | X | X |
| NAMD (MPI) | | X | X |

(**bold X** : already deployed)

(regular X : to be deployed in a second phase)

3.4 - Setting up the environment for the portal applications

The module files

In order to achieve standardization and a rational management of the software versions between the DEISA sites, JRA4 built a number of scripts to create a **deisa portal** module, in collaboration with SA4 and in compliance with the Common Product Environment module files.

The steps to create the **deisa_portal** module are the following :

1) Creation of the directory :

/usr/local/pub/Modules/modulefiles/init_portal that has to contain the file '**deisa_portal**':

```
#%Module - DEISA PORTAL Common Production Environment
# deisa_portal modulefile (IBM AIX systems)
#
# Author          : IDRIS-CNRS
# Created the     : Tue Oct 19 17:25:26 2004
# Last modification by : IDRIS-CNRS
# Last modification the : Thu Mar 29 12:16:59 2007
set ModuleVersion "1.0"
module-whatis "Initializations of the DEISA PORTAL environment."
#### Lines to change depending of the installation
setenv MODULES_BASE "/usr/local/pub/Modules"
setenv MODULEFILES "/usr/local/pub/Modules/modulefiles"

# To do not use any more the special directory used to initialize
# the Modules environment, defined in the init/modulerc file
module unuse "/usr/local/pub/Modules/modulefiles-local/init"
module unuse "/usr/local/pub/Modules/modulefiles/init"

# Initialize the environment variable for the scratch file system
setenv DEISA_SCRATCH "$env(TMPDIR)"
#### Lines below are not supposed to be changed
# DEISA PORTAL Common Production Environment
setenv MODULES_PREFIX "DEISA"
# Useful for module load NAMD
setenv DEISA_NOCOMPILER_WRAPPERS 0

# Definition of the set of directories used to find the modulefiles
# modulefiles for applications
module use --append "$env(MODULEFILES)/applications_portal"
module use --append "$env(MODULEFILES)/applications"
# modulefiles for tools
module use --append "$env(MODULEFILES)/tools"

# Initialize the environment variables for the home and data file systems
set GROUP [exec id -g -n]
setenv DEISA_HOME "/deisa/[string range $GROUP 0 2]/home/$GROUP/$env(USER)"
setenv DEISA_DATA "/deisa/[string range $GROUP 0 2]/data/$GROUP/$env(USER)"

# Help procedure: called by "module help deisa_portal"
proc ModulesHelp {} {
    global ModuleVersion
    puts stderr "
```

```
**** DEISA Common Production Environment ****\n
modulefile \"[module-info name]\" - Version $ModuleVersion\n
Initializations of the DEISA environment."
```

```
    return 0
} "
```

2) Creation of the directories :

```
/usr/local/pub/Modules/modulefiles/applications_portal/blast
/usr/local/pub/Modules/modulefiles/applications_portal/raxml
```

with the files '**2.2.15**' in the blast directory :

```
##%Module - Common Production Environment
```

```
# blast/2.2.15 modulefile
#
# Author          : IDRIS
# Created the     : Thu Mar 29 12:23:55 2007
# Last modification by : IDRIS
# Last modification the : Thu Mar 29 12:23:55 2007
```

```
set ModuleVersion "1.0"
```

```
# Inclusion of site local dependencies
source "$env(MODULEFILES)/sitelocal/applications_portal/blast-2.2.15"
```

```
set ITEM "BLAST v2.2.15"
```

```
module-whatism "Set environment variables to enable the usage of $ITEM"
```

```
# Tests of consistency
# -----
# This application cannot be loaded if another blast* modulefile was previously loaded
conflict blast
```

```
# Extend the path
# -----
prepend-path PATH $BLAST_DIRECTORY_OF_EXECUTABLES
```

```
# Definition of variables
# -----
setenv BLAST_BASES $BLAST_DIRECTORY_OF_BASES
setenv BLAST_DATA $BLAST_DIRECTORY_OF_DATA
```

```
if { [module-info mode] != "whatism" } {
    puts stderr "[module-info mode] [module-info name] (PATH, BLAST_BASES, BLAST_DATA)"
}
```

```
# Help procedure: called by "module help <modulefile>"
proc ModulesHelp {} {
    global env ModuleVersion ITEM
```

```

    puts stderr "
**** $env(MODULES_PREFIX) Common Production Environment ****\n
modulefile \"[module-info name]\" - Version $ModuleVersion\n
Set environment variables to enable the usage of $ITEM\n
\tBLAST_BASES is the directory of the blast base\n
\tBLAST_DATA is the directory of the blast data\n"

return 0
}

```

and '2.2.1' in the raxml directory :

```

#%Module - Common Production Environment

# RAxML/2.2.1 modulefile
#
# Author          : IDRIS
# Created the     : 03/07/2007
# Last modification by : IDRIS
# Last modification the : Thu Mar 29 12:23:55 2007

set ModuleVersion "1.0"

# Inclusion of site local dependencies
source "$env(MODULEFILES)/sitelocal/applications_portal/raxml-2.2.1"

set ITEM "RAxML v2.2.1"

module-whatis "Set environment variables to enable the usage of $ITEM"

# Tests of consistency
# -----
# This application cannot be loaded if another raxml* modulefile was previously loaded
conflict raxml

# Extend the path
# -----
prepend-path PATH $RAXML_DIRECTORY_OF_EXECUTABLES

if { [module-info mode] != "whatis" } {
    puts stderr "[module-info mode] [module-info name] (PATH)"
}

# Help procedure: called by "module help <modulefile>"
proc ModulesHelp {} {
    global env ModuleVersion ITEM
    puts stderr "
**** $env(MODULES_PREFIX) Common Production Environment ****\n
modulefile \"[module-info name]\" - Version $ModuleVersion\n"

return 0
}

```

3) Creation of the directory

/usr/local/pub/Modules/modulefiles/sitelocal/applications_portal

with the file '**blast-2.2.15**'

```
# Common Production Environment - blast/2.2.15 IDRIS local dependencies
#
# Author          : IDRIS-CNRS
# Created the     : Thu Mar 29 14:54:11 2007
# Last modification by : IDRIS-CNRS
# Last modification the : Thu Mar 29 14:54:11 2007

set BLAST_DIRECTORY_OF_EXECUTABLES "/usr/local/pub/BLAST/blast-2.2.15/ncbi/bin"
set BLAST_DIRECTORY_OF_BASES      "/deisa/idr/data/portal_db/db/infobiogen"
set BLAST_DIRECTORY_OF_DATA       "/usr/local/pub/BLAST/blast-2.2.15/ncbi/data"

return
```

and '**raxml-2.2.1**'

```
# Common Production Environment - raxml/2.2.1 IDRIS local dependencies
#
# Author          : IDRIS-CNRS
# Created the     : Thu Mar 29 14:54:11 2007
# Last modification by : IDRIS-CNRS
# Last modification the : Thu Mar 29 14:54:11 2007

set RAXML_DIRECTORY_OF_EXECUTABLES "/usr/local/pub/RAXML/RAXML-VI-HPC-
2.2.1/bin"

return
```

The **deisa_portal** module is only visible to the generic account : idr001ip, the following line has to be added in the .bash_profile file :

"module use --append /usr/local/pub/Modules/modulefiles/init_portal/"

```
zahir001-idr001ip : module avail
----- /usr/local/pub/Modules/modulefiles-local/init -----
xlc/7.0.0.5      xlf/9.1.0.7      petsc/2.3.0
xlc/new         xlf/old          netcdf/3.5.0
xlc/old         xlf/9.1.0.5     netcdf/3.6.1(default)
xlc/8.0.0.12(default) xlf/new          imsl/5.0(default)
xlf/10.1.0.3(default) petsc/2.1.3(default) imsl/4.01
----- /usr/local/pub/Modules/modulefiles/init -----
deisa
----- /usr/local/pub/Modules/modulefiles/init_portal/ -----
deisa_portal
```

Once **deisa_portal** module is loaded, the three portal applications **blast**, **raxml** et **namd** can also be loaded.

- 1) For blast : module load deisa_portal blast
- 2) For raxml : module load deisa_portal raxml
- 3) For namd : module load deisa_portal namd

Typical job files for the three applications

At the request of NICE and ESA 3, three basic job files where composed by JRA4, to show the way the software are typically run.

BLAST job submission :

```
# @ job_name = blast
# @ job_type = serial
# @ output = $(job_name).$(jobid)
# @ error = $(job_name).$(jobid)
# @ cpu_limit = 4:00:00,3:50:00
# @ data_limit = 9Gb
# @ stack_limit = 1Gb,1Gb
# @ resources = ConsumableCpus(16)
# @ shell = /bin/bash
# @ queue

set -x

module load deisa_portal blast

# --- 'blastall' requirements
export BLASTDB=$DEISA_SCRATCH/"
export DATA=$DEISA_SCRATCH
# ---

cd $DEISA_SCRATCH
cp $BLAST_BASES/GoldenPath/hg17* .
cp $BLAST_DATA/* .

# Input File
cp $LOADL_STEP_INITDIR/All.faa0001 .

export OMP_NUM_THREADS=16
blastall -p tblastn -d hg17 -i All.faa0001 -a $OMP_NUM_THREADS -e 1e-4 -o All.out -B 256

cp All.out $LOADL_STEP_INITDIR/.
```

NAMD job submission :

```
# @ job_name = NAMD
# @ job_type = parallel
# @ output = $(job_name).$(jobid)
# @ error = $(job_name).$(jobid)
# @ cpu_limit = 1:00:00,00:55:00
# @ data_limit = 1.1Gb
# @ stack_limit = 0.2Gb,0.2Gb
# @ total_tasks = 20
# @ network.MPI_LAPI = sn_all,shared,US
# @ shell = /bin/bash
# @ queue
```

```
set -x
```

```
module load deisa_portal namd
```

```
cd $DEISA_SCRATCH
```

#Input Files

```
cp $LOADL_STEP_INITDIR/apo* .
cp $LOADL_STEP_INITDIR/par_all22* .
```

```
namd2 apoa1.namd > apoa1.namd.out
```

```
ls -lrt
```

```
cp apoa1.namd.out $LOADL_STEP_INITDIR/.
```

RAxML job submission :

```
# @ job_name = RaxmlMPI
# @ job_type = parallel
# @ output = $(job_name).$(jobid)
# @ error = $(job_name).$(jobid)
# @ cpu_limit = 1:00:00,00:55:00
# @ data_limit = 0.7Gb
# @ stack_limit = 0.2Gb,0.2Gb
# @ total_tasks = 4
# @ network.MPI_LAPI = sn_all,shared,US
# @ shell = /bin/bash
# @ queue
```

```
set -x
```

```
module load deisa_portal raxml
```

```
cd $DEISA_SCRATCH
```

#Input File

```
cp $LOADL_STEP_INITDIR/example_alignment .
```

```
raxmlHPC-MPI -# 4 -s example_alignment -m GTRCAT -n testRaxmlMPI
```

```
cp *.testRaxmlMPI.* $LOADL_STEP_INITDIR/.
```

3.5 - Specification of the user interfaces

The purpose of this activity is to achieve a user friendly interface that will mask the internal complexity of the DEISA supercomputing environment, and will allow life sciences scientist to focus on the scientific part of the work. In the mean time it should be also adaptable to the different skills and knowledge of the user: complex requests or a first level of automation should be feasible.

We are consulting the different sites that propose similar services around the world and we will try to maintain some continuity in the form regarding the web interface, in order not to lose users into a new shiny interface maze. Nevertheless we have also planned some significant added value, for example we want to build a closer relation with the user, and make him able to keep track of the jobs he already passed on the portal.

Databases

For some application like BLAST, that basically queries sequences into a database, the choice of the available databases is a strategic one. To avoid a great number of false positive hits in his query results, the user has to carefully choose the most suitable database to his aims. Therefore we have to be very alert to the user needs in that subject, and flexible, which is not that easy considering that every day new genetic information is found and submitted.

There are three major entities that gather all the nucleotidic data produced in the world, Embl in Europe, Genbank in the United States and DDBJ in Japan. Those banks are exchanging their data on a daily basis which means that they are equivalent. We will download our databases from the EMBL from the EBI (European Bioinformatics Institute) and have it updated every day. For the proteic database our source will be also the database maintained by the EBI : TrEMBL. That first main database represents a set of roughly 500 Gb of data. An important aspect is also the fact that users need specific subsets of theses banks or also other banks that are more accurate, more documented for their species of interest.

RAxML and NAMD are much less demanding in database resources.

Databases will be stored on the GPFS in order to have them accessible by each center, to limit the space and version problems. Also, if for performances motives databases have to be local to a center it will not be a problem with the support of the module command that not only sets the path to the applications but also the path to the database.

Dissemination

We also have lightly initiated some external communication about the portal in the life sciences community, advertising the added value that portal is supposed to provide to very demanding simulations in this area, and searching for beta testers.

4 - Perspectives

In the first phase of the project only the pilot sites are concerned. Here is a table about the future deployment of the project.

| Site | Application | Computing Nodes Architecture | Submission Host Architecture | Scheduler | EF Agent / SSH Plugin |
|---------------|--------------------|-------------------------------------|-------------------------------------|------------------|------------------------------|
| BSC* | NAMD | IBM Power PC Linux | i586 SUSE Linux 10 | SLURM/MOAB | SSH |
| IDRIS* | BLAST | IBM Power4 AIX | IBM Power4 AIX | LoadLeveler | Agent |
| LRZ* | RAxML | SGI Altix Linux | Intel Itanium 2 SUSE Linux 9 | PBSPRO | Agent |
| CINECA | BLAST | IBM Power4 AIX | IBM Power 4 AIX 5.3 | LoadLeveler | SSH |
| FZJ | BLAST | IBM Power4+ AIX | IBM Power 4+ AIX 5.3 | LoadLeveler | SSH |
| RZG | BLAST | IBM Power4 AIX | IBM Power 4 AIX 5.3 | LoadLeveler | Agent |
| EPCC | BLAST | IBM Power5 AIX | Power 5 AIX 5.3 | LoadLeveler | SSH |
| CSC | NAMD | CRAY XT4 Opteron Unicos | Opteron Linux Suze 9 | PBSPRO | Agent |
| SARA | ? | IBM Power 5 Linux | Power 5 - Linux Suze 9 | LoadLeveler | Agent |

* Indicates a pilot site